UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/808,469	03/14/2001	Peter Warnes	ARC.015A	3548

27299 7590 01/06/2005

GAZDZINSKI & ASSOCIATES
11440 WEST BERNARDO COURT, SUITE 375
SAN DIEGO, CA 92127

EXAMINER	
STEELMAN, MARY J	

ART UNIT	PAPER NUMBER
2122	

DATE MAILED: 01/06/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/808,469	WARNES, PETER	
	Examiner	Art Unit	
	Mary J. Steelman	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *02 August 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-17 and 19-41* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-17 and 19-41* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *02 August 2004* is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to Amendment and Remarks filed 2 August 2004. Per Applicant's request, claims 1, 5, 6, 8, 9, 14, 19, 20, 25, 26, 28, and 29 have been amended. Claim 18 has been canceled. New claims 32-41 have been added. Claims 1-17, 19-41 are pending.

Drawings

2. Formal drawings were received for FIG. 2, FIG. 3, and FIG. 33. The drawings are objected to because they lack "REPLACEMENT SHEET" in the top margin.. Corrected drawing sheets in compliance with 37 CFR 1.121(d) are required in reply to the Office action to avoid abandonment of the application. Any amended replacement drawing sheet should include all of the figures appearing on the immediate prior version of the sheet, even if only one figure is being amended. The figure or figure number of an amended drawing should not be labeled as "amended." If a drawing figure is to be canceled, the appropriate figure must be removed from the replacement sheet, and where necessary, the remaining figures must be renumbered and appropriate changes made to the brief description of the several views of the drawings for consistency. Additional replacement sheets may be necessary to show the renumbering of the remaining figures. The replacement sheet(s) should be labeled "Replacement Sheet" in the page header (as per 37 CFR 1.84(c)) so as not to obstruct any portion of the drawing figures. If the changes are not accepted by the examiner, the applicant will be notified and informed of any required corrective action in the next Office action. The objection to the drawings will not be held in abeyance.

Specification

5. In view of the removal of the Appendixes, the prior objection to the Specification is hereby withdrawn. Per Applicant's request, the Specification has been amended.

Claim Objections

6. In view of the amendment to claim 5, the prior objection is hereby withdrawn.

Claim Rejections - 35 USC § 112

4. In view of Applicant's amendments and comments, the prior 35 USC 112 second paragraph rejections are hereby withdrawn.

7. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

8. Claims 36 – 38 are rejected under 35 U.S.C. 112, second paragraph.

9. Claim 36 recites the limitation "the 'n'", in line 22, page 48, of Amendment. There is insufficient antecedent basis for this limitation in the claim. Additionally, it is unclear what the 'n' most significant bits would be. Examiner suggests using the phrase as amended in claim 28: "...for the compressed instruction within a predetermined number of the most significant bits of an instruction word."

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person

having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. Claims 1-41 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent 6,408,428 B1 to Schlansker et al., in view of US Patent 6587939 B1 to Takano.

Claim Rejections - 35 USC § 102

11. Claims 1-11, 14-17, and 19-26 are rejected under 35 U.S.C. 102(e) as being anticipated by US Patent 6,408,428 B1 by Schlansker et al.

Schlansker disclosed a "system as well as methods for automated design" (col. 88, line 23) and a computer readable medium (col. 92, lines 7-8) for development of compressed and optimized instruction sets for processors.

Per claim 1:

-(i) providing a program having a plurality of different instruction types;

(Col. 4, lines 59-61, "Parameterized by this MDES, a re-targetable compiler generates operation issue statistics for a program (program with different instruction types) executing on a candidate processor.", col. 16, lines 45-49, "...programs (program with different instruction types) as well as the input and output data structures are implemented in software stored on the workstations memory system...")

-(iv) creating a compressed instruction set encoding to generate a compressed instruction set based at least in part on said act of determining.

(Col. 3, lines 52-54, "The method then programmatically generates a description of the new candidate processor in a hardware description language from the new specification.", col. 4, lines 42-48, "The instruction format designer programmatically generates an instruction format from the datapath specification and the abstract instruction set architecture specification. This instruction format includes instruction templates representing VLIW instructions executable in the VLIW processor, instruction fields of each of the templates, and bit positions and encoding for the instructions fields", col. 14, lines 10-12, "The role of spacewalker is to identify processors (identifies processor designs) which deliver the greatest possible performance at the lowest possible cost.", (compressed / optimized), col. 18, lines 40-46, "The instruction format specifies the instructions capable of being executed in a VLIW processor design. These instructions are represented as instruction templates...also includes the instruction fields within each template, and the bit positions and encodings for the instruction fields", col. 22, lines 20-21, "...the processor design includes the instruction format specification...")

Schlansker failed to specifically disclose:

-(ii) determining the static frequency of each of said instruction types from a base instruction set;

-(iii) determining the number and type of instructions necessary for correct instruction set execution based at least in part on said act of determining the static frequency;

However, Takano disclosed (col. 2, lines 18-27) an "invention to provide an information processing apparatus having a compact storage region required of an executable program with a reduced number of instructions constituting the executable program...characterized by provision of a executable instruction extracting unit for extending a compressed instruction into a plurality of corresponding executable instructions, and a reconfiguration unit for reconfiguring and optimizing the executable instruction extracting unit suitable for the executable program."

See FIG. 9, #401: the instruction sequence occurrence frequency detecting unit. Col. 7, lines 20-23, "The instruction sequence occurrence frequency detecting unit serves to obtain the occurrence frequency of the instruction sequences contained in the assembler source file (determining the static frequency of each of said instruction types from a base instruction set) ..." The instruction frequency is detected and (col. 7, line54-65) "registered in the compressed/executable instruction correspondence table (by type)..." Col. 8, lines 11-14, "...under the condition that the result of execution of the executable program is not influenced by the modification (necessary for correct instruction set execution) ..."

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including static frequency of instructions when creating a compressed instruction set as both inventions are directed towards optimal processor configuration for a target program,

resulting in reduced storage requirements (Takano, col. 2, lines 29-31.) Schlansker, col. 1, line 55, referenced the desire for "optimizing architectures for particular application domains."

Per claim 2:

-re-evaluating said compressed instruction set using at least said steps (i), (ii), and (iii);

Schlansker disclosed the "instruction format" as a part of the processor design. The candidate processor is evaluated and repeated, using metrics to optimize. Schlansker disclosed (col. 3, lines 65-67), "The process of specifying and evaluating candidate processors may be repeated to explore the parameterized design space.", col. 6, lines 58-59, "This process is iterated until systems of adequate quality are identified.", col. 22, lines 23-26, "One form of optimization used in the system is the customization or optimization of the processor design based on internal usage statistics..."

More specifically, Takano disclosed compressing and extending an instruction set / format, in order to reduce storage requirements. Additionally, Takano disclosed repeating the compression technique as needed at col. 13, lines 21-27.

-generating an instruction set encoding for said compressed instruction set.

Schlansker disclosed, (col. 22, lines 31-36), "The MDES extraction process programmatically generates an MDES description for driving a retargetable compiler. The MDES is represented in database tables that provide the op code repertoire of the processor, their IO formats, their

latencies and resource sharing constraints of the operations...", col. 28, line 66- col. 29, line 2, "The abstract ISA spec is a machine-readable data structure that specifies register files, operation groups, ILP constraints, and architecture parameters." He failed to specifically disclose 'generating an instruction set encoding for said compressed instruction set.'

However, Takano explicitly disclosed (col. 3, lines 51-54), "The executable instruction extracting unit reconfiguration unit serves to reconfigure the executable instruction extracting unit on the basis of the compressed/executable instruction correspondence table." Takano, col. 7, lines 56-65, making use of frequency counts, instructions are compressed and registered in the compressed/executable instruction correspondence table. See FIG. 10. Thus, Takano disclosed generating an encoding for the compressed instruction set.

-regarding "static frequency" as referenced in steps ii and iii of claim 1 above. Takano disclosed this feature as noted in the rejection of claim 1 above.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including static frequency of instructions when creating/ generating a compressed instruction set, and re-evaluating, as both inventions are directed towards optimal processor configuration for a target program, resulting in reduced storage requirements (Takano, col. 2, lines 29-31.) Schlansker, col. 1, line 55, referenced the desire for "optimizing architectures for particular application domains."

Per claim 3:

-the act of providing a program comprises providing an assembly language program.

Schlansker disclosed (Col. 16, lines 45-49), "...programs as well as the input and output data structures are implemented in software stored on the workstation's memory system...may be implemented using standard programming languages...", but failed to specify "assembly language." However, Takano disclosed "assembler source file" at col. 7, line 15 and FIG. 9.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including specifically noting that assembly language could be used as the source input because Schlansker broadly suggested (col. 16, lines 45-49.) the source could be a "standard programming language."

Per claim 4:

-sorting said instruction types by frequency of usage.

Schlansker disclosed (col. 7, line 41-45), "The re-targetable compiler schedules an application program and generates a number of statistics...The operation issue statistics provide histograms indicating the static and dynamic opcodes usage...(sort by frequency)", broadly disclosing that statistics are collected. However, Takano more explicitly disclosed the sort. See FIG. 9 which shows a count per instruction.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including more details related to frequency counts of instructions when creating/ generating a compressed instruction set as both inventions are directed towards optimal processor configuration for a target program, resulting in reduced storage requirements (Takano, col. 2, lines 29-31.) Schlansker, col. 1, line 55, referenced the desire for "optimizing architectures for particular application domains."

Per claim 5:

-digital processor includes an extension logic unit adapted to execute at least one extension instruction, and the act of providing comprises providing a program having said at least one extension instruction, said at least one execution instruction being executable by said extension logic unit.

Schlansker disclosed, (col. 23, lines 2-22), "In the first case, the datapath specification may have been specified by hand...In the second case, the concrete ISA specification may be provided as input based on some existing processor design...Alternatively, the developer may wish to optimize an existing concrete ISA specification for a particular application or application program (extension logic). To support these design scenarios, the system includes modules for extracting an abstract ISA specification...and concrete ISA specification respectively...may alter the abstract ISA specification...One particular example is the use of custom templates based on

operation issue statistics...may alter the opcodes repertoire and ILP constraints to achieve an optimized design based on cost/performance trade-offs." Also see col. 3, lines 32-34, "A compiler, re-targeted to the candidate processor, generates operation issue statistics for an application program to be executed in the candidate processor.")

Takano explicitly disclosed extension logic (col. 5, lines 8-9), "...in the case of a compressed instruction, it is extended into the corresponding executable instructions..."

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including more details related to executable extension logic when creating/ generating a compressed instruction set as both inventions are directed towards optimization. Schlansker (col. 1, line 55) referenced the desire for "optimizing architectures for particular application domains."

Per claim 6:

-the act of creating a compressed instruction set comprises selecting "N" instructions having the greatest frequency of occurrence, said selected "N" instructions permitting said program to be compiled with a predetermined size.

Schlansker disclosed giving consideration to code size. (Col. 83, line 54-57) "The compiled application is assembled and linked to determine the application's code size (and ROM area) as well as an estimate of the number of cycles needed to execute it..."

Takano provided more details related to selecting instructions with the greatest frequency of occurrence. See FIG. 9 for frequency count of instructions. Col. 9, lines 30-34, "In the case that the requirement of defining compressed instructions is the occurrence frequency of the corresponding sequence or executable instruction is determined as no smaller than 2..." Thus Takano suggested that a specified count (N) is used to determine which instructions are chosen to be a candidate for compression.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including more details related to executable extension logic when creating/ generating a compressed instruction set as both inventions are directed towards optimization. Schlansker (col. 1, line 55) referenced the desire for "optimizing architectures for particular application domains."

Per claim 7:

-the act of determining a compression ratio for said compressed instruction set, said compression ratio being related to the ratio of the number of compressed instructions to the total number of original instructions.

(Col. 7, line 57 – col. 8, line 3, "The system includes a program or programs that implement search heuristics to select candidate processor designs for evaluation. These search heuristics use information about a candidate processor's cost and performance to select other candidates. A performance evaluator computes the performance of a candidate processor in terms of execution cycles. A cost evaluator evaluates the cost of a candidate processor based on costs of individual components in the hardware description, which lists instances of macrocells and their corresponding areas, power consumption, etc...internal resource usage information (compression ration) may be used to refine or focus the search more effectively.")

Schlansker did not explicitly disclose a "compression ratio". However, he did disclose that heuristics and internal resource usage information are involved in the decision for an optimal design.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Schlansker's invention, to broadly interpret the consideration of heuristics and internal resource usage information as including the ratio of compressed instructions to original instructions, as all heuristics are meant to optimize processor instruction set design.

Per claim 8:

-a processor core having an instruction pipeline comprising at least instruction fetch, decode, and execute stages;

(Col. 16, lines 8-12, "Software pipelining is a compile-time scheduling technique...to speed up execution...", col. 20, lines 36-39, "...components may include a prefetch buffer that covers the latency of sequential instruction fetch, an instruction register for storing the next instruction to be issued to the decode logic...", col. 20, lines 62-67, "...code simulators can simulate the program...and provide statistics...during execution...")

-a data interface in data communication with said processor core, said interface adapted for data communication with a storage device configured to hold a plurality of program instructions;

(Col. 19, lines 54-59, "The instruction unit includes a control path and an instruction sequencer...the control logic for controlling the IUdatapath...the instruction decode logic for decoding each instruction...", col. 20, lines 42-44, "...specifies the method for fetching instruction from an instruction cache and issuing them to the control ports in the data path (data communication)", col. 21, lines 36-44, "The design flow...begins with an abstract ISA specification...The system may then create an instruction format specification based on the data path and abstract ISA spec...")

-an optimized instruction set comprising a base instruction set and a compressed instruction set, said compressed instruction set being generated by the method comprising:

(Col. 23, lines 16-19, "Once the abstract ISA specification is extracted (base instructions), the system...may alter the abstract ISA specification (optimized / compressed instruction set)...use of custom templates based on operation issue statistics...")

-determining the static frequency of each of said instruction types from said base instruction set;

(See limitations addressed in claim 1.)

-determining the number and type of instructions necessary for instruction set execution based at least in part on said act of determining the static frequency;

(See limitations addressed in claim 1.)

-creating a compressed instruction set encoding to generate said compressed instruction set.

(See limitations addressed in claim 1.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including static frequency of instructions when creating a compressed instruction set

as both inventions are directed towards optimal processor configuration for a target program, resulting in reduced storage requirements (Takano, col. 2, lines 29-31.) Schlansker, col. 1, line 55, referenced the desire for "optimizing architectures for particular application domains."

Per claim 9:

-the act of creating a compressed instruction set comprises selecting "N" instructions having the greatest frequency of occurrence, said selected "N" instructions permitting said program to be compiled with a predetermined size.

(See limitations addressed in claim 6.)

Per claim 10:

-said optimized instruction set also comprises at least one extension instruction adapted to perform a predetermined function, said processor further comprises an extension logic unit adapted to execute said at least one extension instruction.

(Col. 23, lines 2-22, "...optimize an existing concrete ISA specification for a particular application or application program (ASIC)...extracting an abstract ISA specification from an existing datapath and concrete ISA specification...alter the abstract ISA specification...One particular example is the use of custom templates based on operation issue statistics...")

Per claim 11:

-an encoding structure having an opcode and a plurality of instruction slots.

(Col. 52, lines 57-60, "The instruction format assigns sets of op groups (encoding structure having an opcodes)...to slots of an instruction (plurality of instruction slots). The processor issues operations within an instruction from these slots concurrently...", col. 53, lines 44-46, "...each template is partitioned into one or more operation issue slots. Every combination of operations assigned to these slots may be issued concurrently.")

Per claims 12 and 13:

-plurality of instruction slots comprise two slots, each of said slots having two 14-bit instructions / encoding structure comprises 32-bits, and said opcode is disposed within the last four bits thereof.

(Col. 52, lines 24-33, "VLIW processors issue instructions having multiple instruction fields. An instruction field is a set of bit positions intended to be interpreted as an atomic unit within some instruction context. Familiar examples are opcodes fields, source and destination register specifier fields, and literal fields. Bits from each of these fields flow from the instruction register to control ports in the data path..." VLIW allow for variable length and number of slots (col. 54, line 59). The number of slots, bits per slot, the placement of the opcodes is a design choice. See col. 54, lines 21-26. Also, col. 49, lines 53-58, "...the instruction format assigns bit positions and encodings to each of them. The bit positions are specific positions that each field occupies in an instruction. The encodings are the binary values associated with the instruction fields...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention, allowing for specified bit locations, and instruction fields, because it is merely a design choice.

Per claim 14:

-a processor core having a pipeline comprising at least instruction fetch, decode, and execute stages;

(See limitations addressed in claim 8. Also, col. 16, lines 8-12, "...pipelining...")

-a memory interface adapted to at least read program instructions from a program memory and provide said instructions to said pipeline; (See limitations in claim 8. Also, col. 16, lines 45-47, "...programs as well as the input and output data structures are implemented in software stored on the workstation's memory system (memory interface)..."

-an optimized instruction set comprising a base instruction set and a compressed instruction set, said compressed instruction set having a predetermined number and type of instructions necessary for correct instruction set execution on said processor core, said predetermined number and type based at least in part on the static frequency of occurrence of instructions within said base instruction set.

(See limitations addressed in claim 8.)

Per claim 15:

-an encoding structure having an opcode and a plurality of instruction slots.

(See limitations addressed in claim 11.)

Per claim 16:

-said instruction set includes at least one extension instruction, said at least one extension instruction adapted to perform a predetermined function upon execution within said processor.

(Col. 51, lines 38-43, "To optimize an iformat design for a particular application program, the iformat system selects custom templates from operation issue statistics obtained from scheduling the program. The iformat system then generates an iformat based on a combination of the custom templates and an abstract ISA specification.")

Per claim 17:

-an extension logic unit adapted to execute said at least one extension instruction.

(See limitations addressed in claim 5.)

Per claim 19:

A method of enhancing the performance of a reduced instruction set processor, said processor having a multi-stage instruction pipeline and an instruction set having at least a base instruction set, said base instruction set having a plurality of instruction types associated therewith; comprising;

(Col. 16, line 38, Schlansker disclosed "RISC / reduced instruction set processor" Also, see limitations of claims 1 and 8.)

-providing a program having a plurality of instructions;

(See limitations addressed in claim 1.)

-determining the static frequency of each of said instruction types within said plurality of instructions of said program;

(See limitations addressed in claim 1.)

-selecting a number instructions having the greatest frequency of occurrence, said selected instructions allowing said program to be compiled with a predetermined size;

(See limitations addressed in claims 1, 6 & 9.)

-compiling said program based at least in part on said selected instructions.

(See limitations addressed in claims 6 & 9.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the performance enhancement of a reduced instruction set processor, by including static frequency of instructions when creating a compressed instruction set as both inventions are directed towards optimal processor configuration for a target program, resulting in reduced storage requirements (Takano, col. 2, lines 29-31.) Schlansker, col. 1, line 55, referenced the desire for "optimizing architectures."

Per claim 20:

An application specific integrated circuit (ASIC) comprising:

(Col. 6, lines 33-36, "...invention is...system for automating the design of embedded systems..." ASIC is and example of an embedded system.)

-a first processor core, said processor core having a pipeline with at least instruction fetch, decode, and execute stages associated therewith;

(See limitations addressed in claims 1 & 8.)

-an optimized instruction set comprising a base instruction set and a compressed instruction set, said compressed instruction set having a predetermined number and type of instructions, said predetermined number and type based at least in part on the static frequency of occurrence of instructions within one extension instruction adapted to perform at least one specific operation;

(See limitations addressed in claim 8.)

-at least one storage device adapted to store a plurality of data bytes therein, said at least one storage device being accessible by said first processor core;

(Col. 20, lines 42-44, "Specifies the method for fetching instructions from an instruction cache (storage device) and issuing them to the control ports (accessible to processor) in the data path.")

-at least one extension logic unit adapted to facilitate execution of said at least one execution instruction.

(See limitations of claim 5.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including static frequency of instructions when creating a compressed instruction set and by including more details related to executable extension logic when creating/ generating a compressed instruction set, as both inventions are directed towards optimization. Schlansker (col. 1, line 55) referenced the desire for "optimizing architectures for particular application domains."

Per claim 21:

-a second processor core, said second processor core being disposed on the same die as said first processor core.

(See col. 29, Functional Unit Synthesis. Col. 31, lines 27-29, "Build a valid list of functional units...that will support the opcodes and latency requirements of each of the operation groups." A functional unit could be a second processor. See FIG. 11, 472A & 472B.)

Per claim 22:

-said second processor core comprises a digital signal processor (DSP), said DSP being adapted to perform at least one operation on data provided to said ASIC.

(A functional unit could be a DSP. Col. 46, line 62-col. 47, line9, "The data inputs of functional units...are connected to output (read) ports...and the register file, via interconnect buses...The

opcodes repertoire of functional unit includes opcodes LAND, IADD (adapted to perform at least one operation on data); input data for these opcodes is supplied at functional unit input ports...The output of the functional unit...is driven onto the interconnect bus by a tristate buffer in response to a control input...")

Per claim 23:

-said DSP is adapted for initiation by an instruction from said first processor core.

(Col. 45, line 40-col. 46, line 2, "The MDES may also be extracted from the abstract ISA specification provided in the ArchSpec...The ILP constraints can be used to extract abstract resource constraints needed to re-target the compiler...This is still useful, for example, in application specific processor design where a quick retargeting of the compiler is needed to arrive at a desired abstract instruction set architecture of customized instruction templates...")

Per claim 24:

-at least a portion of the operation of said DSP is controlled by extension registers associated with said first processor core.

(Col. 69, lines 61-63, "...the data path input specifies instances of the functional unit macrocells and register file macrocells in the data path...", col. 70, lines 8-11, "The data path includes a register file instance, gpr, (extension registers) a functional unit (FU) (DSP)cell instance, and an interconnect (association) between the gpr and functional unit.")

Per claim 25:

A pipelined digital processor (See limitations addressed in claim 8.), comprising:

-processor means having an instruction pipeline comprising at least means for instruction fetch, means for instruction decode, and means for instruction execution;

(See limitations addressed in claim 8.)

-means for data interface, said means for data interface being in data communication with said processor core, said means for data interface adapted for data communication with a storage device configured to hold a plurality of program instructions;

(See limitations addressed in claim 8.)

-optimized instruction means comprising base instruction means and compressed instruction means, said compressed instruction means being generated by the method comprising:

(See limitations addressed in claim 8.)

-determining the static frequency of each of said instruction types from said base instruction means;

(See limitations addressed in claims 1 & 8.)

-determining the number and type of instructions necessary for instruction execution based at least in part on said act of determining the static frequency;

(See limitations addressed in claims 1 & 8.)

-creating a compressed instruction encoding to generate said compressed instruction means.

(See limitations addressed in claims 1 & 8.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including static frequency of instructions when creating a compressed instruction set as both inventions are directed towards optimal processor configuration for a target program, resulting in reduced storage requirements (Takano, col. 2, lines 29-31.) Schlansker, col. 1, line 55, referenced the desire for "optimizing architectures for particular application domains."

Per claim 26:

A method of operating an extended pipelined digital processor having an instruction pipeline comprising at least instruction fetch, decode, and execute stages, a storage device configured to hold a plurality of program instructions, and an optimized instruction set, the method comprising:

(See limitations addressed in claim 8.)

-providing a base instruction set having a plurality of instructions;

(See limitations in claim 8.)

-providing an extension instruction set determined at least in part by selections made by a user;

Schlansker disclosed, (col. 23, lines 2-22), "In the first case, the datapath specification may have been specified by hand...In the second case, the concrete ISA specification may be provided as input based on some existing processor design...Alternatively, the developer may wish to optimize an existing concrete ISA specification for a particular application or application program (extension logic). To support these design scenarios, the system includes modules for extracting an abstract ISA specification...and concrete ISA specification respectively...may alter the abstract ISA specification...One particular example is the use of custom templates based on operation issue statistics...may alter the opcodes repertoire and ILP constraints to achieve an optimized design based on cost/performance trade-offs." Also see col. 3, lines 32-34, "A compiler, re-targeted to the candidate processor, generates operation issue statistics for an application program to be executed in the candidate processor.")

Takano explicitly disclosed extension logic (col. 5, lines 8-9), "...in the case of a compressed instruction, it is extended into the corresponding executable instructions..."

-providing a compressed instruction set derived at least in part from said base and extension instruction sets;
(See limitations in claim 8.)

-assigning one of a plurality of predetermined values to at least one bit within a status register within said processor;

(Col. 15, lines 27-30, "Operations read an additional guarding predicate operand, typically a single bit stored in a predicate register file. These operations either execute or are nullified (plurality of values) according to the value of the guarding predicate (status register).")

-executing at least one instruction from said base instruction set within said pipeline based on a first predetermined value present in said status register;

(Use the "concrete ISA", see col. 23, line 14.)

-executing at least one instruction from said compressed instruction set within said pipeline based on a second predetermined value present in said status register.

(Execute using an altered abstract ISA (see col. 23, lines 13-19) and customized template.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including more details related to executable extension logic when creating/ generating a compressed instruction set as both inventions are directed towards optimization. Schlansker (col. 1, line 55) referenced the desire for "optimizing architectures for particular application domains."

Per claim 27:

-the act of assigning comprises assigning a '1' or '0' value to a low address (L) bit within said register.

(Schlansker disclosed the design of processors, including the instruction sets. Schlansker failed to disclose various design choices such as the number of bits per slot, chosen bit locations (LSB, MSB) for various instruction fields. Col. 15, lines 12-14, "In control speculation, specialized hardware in the processor uses tagged operands ('1' or '0') to track erroneous or exceptional results..." Also, regarding the assignment of bit positions, see col. 21, lines 52-57, "It then builds the IF-tree data structure containing instruction fields...that need to be assigned bit positions within each instruction template. To set up the bit allocation problem, ...partitions instruction fields into groups based on instruction filed to control port mappings..." The location of the bits in the instruction is a design choice.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention, allowing for specified bit locations, and instruction fields, because it is merely a design choice.

Per claim 28:

-the act of providing a compressed instruction set comprises encoding at least a portion of the instruction operation codes (opcodes) for the compressed instructions within a predetermined number of the most significant bits of an instruction word.

(Col. 21, lines 52-57, "It then builds the IF-tree data structure containing instruction fields...that need to be assigned bit positions within each instruction template. To set up the bit allocation

problem, ...partitions instruction fields into groups based on instruction filed to control port mappings..." The location of the bits in the instruction is a design choice.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention, allowing for specified bit locations, and instruction fields, because it is merely a design choice.

Per claim 29:

-the act of providing a compressed instruction set further comprises encoding a plurality of said compressed instruction with source register fields located in a predetermined relationship to one another within said compressed instructions.

(Col. 52, lines 24-37, "VLIW processors issue instructions having multiple instruction fields. An instruction field is a set of bit positions intended to be interpreted as an atomic unit within some instruction context. Familiar examples are opcodes fields, source and destination register specifier fields, and literal fields. Bits from each of these fields flow from the instruction register to control ports...opcodes bits flow to functional units and source register its flow to register file read address ports...", col. 60, line 15-19, "The read/write ports of various register files in the datapath need to be provided address bits to select the register to be read or written. The number of bits needed for these fields depends on the number of registers...")

Takano disclosed more specific details related to compressed instructions. Col. 2, lines 18-27, an "invention to provide an information processing apparatus having a compact storage region required of an executable program with a reduced number of instructions constituting the executable program...characterized by provision of a executable instruction extracting unit for extending a compressed instruction into a plurality of corresponding executable instructions, and a reconfiguration unit for reconfiguring and optimizing the executable instruction extracting unit suitable for the executable program."

See FIG. 9, #401: the instruction sequence occurrence frequency detecting unit. Col. 7, lines 20-23, "The instruction sequence occurrence frequency detecting unit serves to obtain the occurrence frequency of the instruction sequences contained in the assembler source file (determining the static frequency of each of said instruction types from a base instruction set) ..." The instruction frequency is detected and (col. 7, line54-65) "registered in the compressed/executable instruction correspondence table (by type)..." Col. 8, lines 11-14, "...under the condition that the result of execution of the executable program is not influenced by the modification (necessary for correct instruction set execution) ..."

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including static frequency of instructions when creating a compressed instruction set as both inventions are directed towards optimal processor configuration for a target program,

resulting in reduced storage requirements (Takano, col. 2, lines 29-31.) Schlansker, col. 1, line 55, referenced the desire for "optimizing architectures for particular application domains."

Per claim 30:

-the act of encoding with said predetermined relationship comprises encoding the source register fields for respective ones of said plurality of compressed instructions at identical locations.

(Col. 52, lines 30-37, "...source register bits flow to register file read address ports...", col. 63, lines 48-52, "...two non-conflicting fields that map to the same register file read address port. By assigning a single set of bit positions to the two fields...we attempt to allocate the same bit positions to affinity siblings.")

Per claim 31:

-the act of providing a compressed instruction set comprises encoding all of the immediate data fields such that they start from the least significant bit (LSB).

(Schlansker disclosed the design of processors, including the instruction sets. Schlansker failed to disclose various design choices such as the number of bits per slot, chosen bit locations (LSB, MSB) for various instruction fields. Col. 60, lines 21-23, "Some operation formats specify an immediate literal operand (immediate data field) that is encoded within the instruction. The width of these literals is specified externally in the ArchSpec..." The placement of the bits is a design consideration.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention, allowing for specified bit locations, and instruction fields, because it is merely a design choice.

Per claim 32:

A user-configured and extended pipelined RISC processor, comprising:

-a processor core having an instruction pipeline comprising at least instruction fetch, decode, and execute stages;

(Refer to the rejection of claim 26 regarding "user-configured and extended pipelined" and claim 19 regarding RISC / reduced set instruction set processor.)

-a data interface in data communication with said processor core, said interface adapted for data communication with a storage device configured to hold a plurality of program instructions;

(See rejection of limitations as addressed in claim 8 above.)

-an optimized instruction set comprising a base instruction set and a compressed instruction set, said compressed instruction set being generated by the method comprising:

(See rejection of limitations as addressed in claim 8 above.)

-determining the static frequency of each of said instruction types from said base instruction set;

(See rejection of limitations as addressed in claims 1 & 8 above.)

-determining the number and type of instructions necessary for instruction set execution based at least in part on said act of determining the static frequency;

(See rejection of limitations as addressed in claims 1 & 8 above.)

-creating a compressed instruction set encoding to generate said compressed instruction set, said creating comprising using an encoding structure having a opcodes and two instruction slots, each of said slots having two 14-bit instructions, and selecting two instructions having the greatest frequency of occurrence, said selected two instructions permitting said program to be compiled with a predetermined size.

(See rejection of limitations as addressed in claims 6, 9 & 12 above.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including more details related to executable extension logic when creating/ generating a compressed instruction set as both inventions are directed towards optimization. Schlansker (col. 1, line 55) referenced the desire for "optimizing architectures for particular application domains."

Per claim 33:

-said structure comprises a 32-bit encoding structure with said opcodes disposed within the last four bits thereof.

(See rejection of limitations as addressed in claim 13 above.)

Per claim 34:

-A user-configured and extended pipelined RISC processor, comprising:

(See rejection of limitations as addressed in claim 32 above.)

-a processor core having an instruction pipeline comprising at least instruction fetch, decode, and execute stages;

(See rejection of limitations as addressed in claim 32 above.)

-a data interface in data communication with said processor core, said interface adapted for data communication with a storage device configured to hold a plurality of program instructions;

(See rejection of limitations as addressed in claim 32 above.)

-an optimized instruction set comprising a base instruction set and a compressed instruction set, said compressed instruction set being generated by the method comprising:

(See rejection of limitations as addressed in claim 32 above.)

-determining the static frequency of each of said instruction types from said base instruction set;

(See rejection of limitations as addressed in claim 32 above.)

-determining the number and type of instructions necessary for instruction set execution based at least in part on said act of determining the static frequency;

(See rejection of limitations as addressed in claim 32 above.)

-creating a compressed instruction set encoding to generate said compressed instruction set, said creating comprising using a 32-bit encoding structure having an opcodes and a plurality of instruction slots, said opcodes being disposed within at least the last 4-bits of said structure, and selecting a corresponding number of instruction having the greatest frequency of occurrence, said selected instruction permitting said program to be compiled with a predetermined size.
(See rejection of limitations as addressed in claims 1, 9, & 13 above.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including more details related to executable extension logic when creating/ generating a compressed instruction set as both inventions are directed towards optimization. Schlansker (col. 1, line 55) referenced the desire for "optimizing architectures for particular application domains."

Per claim 35:

A method of operating a user-extended and configured RISC processor having an instruction pipeline comprising at least instruction fetch, decode, and execute stages, a storage device configured to hold a plurality of program instructions, and an optimized instruction set, the method comprising:
(See rejection of limitations as addressed in claim 20 above.)

-providing a base instruction set having a plurality of instructions and an extension instruction set having at least one extension instruction;

(See rejection of limitations as addressed in claim 20 above.)

-providing a compressed instruction set derived at least in part from said base instruction set;

(See rejection of limitations as addressed in claim 20 above.)

-assigning a value to a compressed instruction selection bit within a status register within said processor;

(See rejection of limitations as addressed in claims 25 & 26 above.)

-executing at least one instruction from said base instruction set within said pipeline based on a first predetermined value present in said status register;

(See rejection of limitations as addressed in claims 25 & 26 above.)

-executing at least one instruction from said compressed instruction set within said pipeline based on said assigned value of said compressed instruction selection bit in said status register.

(See rejection of limitations as addressed in claim 25 above.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor

systems, by including more details related to executable extension logic when creating/ generating a compressed instruction set as both inventions are directed towards optimization. Schlansker (col. 1, line 55) referenced the desire for "optimizing architectures for particular application domains."

Per claim 36:

-the act of providing a compressed instruction set comprises encoding at least a portion of the instruction operation codes (opcodes) for the compressed instructions within the 'n' most significant bits.

(See rejection of limitations as addressed in claim 28 above.)

Per claim 37:

-the act of providing a compressed instruction set further comprises encoding a plurality of said compressed instructions with source register fields located in a predetermined relationship to one another.

(See rejection of limitations as addressed in claim 29 above.)

Per claim 38:

-the act of encoding with said predetermined relationship comprises encoding the source register files for respective ones of said plurality of compressed instructions at identical locations.

(See rejection of limitations as addressed in claim 30 above.)

Per claim 39:

-the act of providing a compressed instruction set comprises encoding all of the immediate data fields such that they start from the least significant bit (LSB).

(See rejection of limitations as addressed in claim 31 above.)

Per claim 40:

A method of operating an extended pipelined digital processor having an instruction pipeline comprising at least instruction fetch, decode, and execute stages, a storage device configured to hold a plurality of program instructions, and an optimized instruction set, the method comprising:

(See rejection of limitations as addressed in claim 8 & 35 above.)

-providing an extension instruction set having a plurality of user-selected extension instructions;

(See rejection of limitations as addressed in claims 5, 8, 10, and 35 above.)

-providing a compressed instruction set derived at least in part from said extension instruction set;

(See rejection of limitations as addressed in claims 1, 8, 20 and 35 above.)

-encoding a plurality of compressed instructions from said compressed instruction set into an instruction word having an op-code.

(See rejection of limitations as addressed in claims 11, 32 & 33 above.)

-assigning one of a plurality of predetermined values to at least one bit within a status register with said processor;

(See rejection of limitations as addressed in claim 27 above.)

-executing at least one of said compressed instructions from said instruction word within said pipeline based on a second predetermined value present in said status register.

(See rejection of limitations as addressed in claim 26 above.)

Per claim 41:

-said act of encoding comprises encoding two 14-bit compressed instructions into a 32-bit aligned instruction word having said opcodes disposed within at least the last four bits thereof.

(See rejection of limitations as addressed in claim 32 above.)

Response to Arguments

3. Applicant's arguments with respect to claims 1-17 and 19-31 have been considered but are moot in view of the new ground(s) of rejection.

Applicant has argued, in substance, the following:

(A) As noted on page 52, last paragraph of Amendment, the prior art fails to teach "a determination of the static or non-operational frequency of instruction types within the code set."

Examiner's Response:

New art has been combined to reject this newly added limitation. Takano has disclosed a static frequency count of instructions. See FIG. 9 and col. 7, lines 15-18. A 'frequency detecting unit' counts the instruction frequency. Instructions to compress are (col. 9, lines 48-52) selected with reference to the occurrence frequencies.

(B) As noted on page 52, last paragraph, Applicant summarizes Schlansker's invention as: "These operation statistics are used to enable subsequent iterations of the hardware design to make use of these relationships. Hence, Schlansker teaches iteratively refining the hardware design of a processor based on operational instruction issue statistics.

Examiner's Response: Schlansker disclosed the "instruction format" as a part of the processor design (col. 4, lines 31-34). The candidate processor is evaluated and repeated, using metrics to optimize. More specifically, Takano disclosed compressing and extending an instruction set / format, in order to optimally reduce storage requirements. Additionally, Takano disclosed repeating the compression technique as needed at col. 13, lines 21-27.

(C) As noted on page 53, 3rd paragraph, Applicant's invention can be used with a varying number of instruction slots. Applicant's invention is intended to be adapted to an existing processor architecture / architecture is not iteratively revised based on the determination of static frequency of instruction types.

Examiner's Response:

Schlansker disclosed an instruction format designer (col. 4, line 32) as one of the components for designing a processor. Col. 4, lines 43-45, "The instruction format designer programmatically generates an instruction format from the datapath specification and the abstract instruction set architecture specification.". Schlansker does not limit the number of instruction slots that may be available. The datapath (col. 4, lines 39-42) "includes instances of functional units, register files...", and thus defines the architecture. The encoding structure is generated to be compatible with the architecture. Takano provided more details on (col. 1, lines 13-14) "extending a compressed instruction", (col. 2, lines 18-20) reducing the number of instructions to accommodate a compact storage region. Col. 5, lines 8-9, "...compressed instruction...is extended into the corresponding executable instructions..."

Conclusion

4. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period

will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

4. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (571) 272-3704. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached at (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman

12/20/2004

TUAN DAM
SUPERVISORY PATENT EXAMINER